

Document Stores

Mit Fokus auf MongoDB

Gliederung

1. Allgemein
2. MongoDB
3. Live Demo
4. Fragen
5. Key Takeaways

Documents

XML, YAML, JSON etc.



```
{  
  "FirstName": "Bob",  
  "Address": "5 Oak St.",  
  "Hobby": "sailing"  
}
```



```
<contact>  
  <firstname>Bob</firstname>  
  <lastname>Smith</lastname>  
  <phone type="Cell">(123) 555-0178</phone>  
  <phone type="Work">(890) 555-0133</phone>  
  <address>  
    <type>Home</type>  
    <street1>123 Back St.</street1>  
    <city>Boys</city>  
    <state>AR</state>  
    <zip>32225</zip>  
    <country>US</country>  
  </address>  
</contact>
```

Document Stores - Victor Wolf

19.05.2020

3

1. The central concept of a document-oriented database is the notion of a *document*. While each document-oriented database **implementation differs** on the details of this definition, in general, they all assume **documents encapsulate and encode data (or information) in some standard format** or encoding. Encodings in use include [XML](#), [YAML](#), [JSON](#), as well as binary forms like [BSON](#).
2. Nähe zum Internet
3. Ähnlichkeit zu einem **Objekt** aus der Objekt orientierten Programmierung
4. Jedes Dokument bekommt eine eindeutige ID mit der es eindeutig identifiziert werden kann.
5. Extrembeispiel Document Store: key aus zahlen mit plain text aber das stimmt nicht ganz
6. Können Metadaten enthalten
7. Sind somit semi strukturiert.
8. Spezialisierte Key-Value Datenbank. Es gibt Felder und Werte diese können sich in der Struktur aber unter den Dokumenten unterscheiden!
9. Durch die Ähnlichkeit zu Objekten direkte Konversion zu Objekten in den meisten Programmiersprachen möglich sowie Polymorphismus

CRUD Operations

Creation

INSERT

Retrieval

GET

Update

UPDATE

Deletion

DELETE

Document Stores - Victor Wolf

19.05.2020

4

- Basis für persistente Datenspeicherung
- Kommt euch das bekannt vor? Tipp: Denkt an das Internet
- Hier sind die Aktionen gemeint die mit der Datenbank abgedeckt werden können -> Mindestanforderung

Datenorganisation

Collections

Tags and
non-visible
Metadata

Directory
Hierachies

Document Stores - Victor Wolf

19.05.2020

5

- Variiert wieder je nach Implementierung, da es kein hartes Datenmodell gibt.
- Collections: Ähnliche Dokumente werden in einer Collection gruppiert Müssen aber nicht den selben Aufbau haben. Ähnlich zu Tabellen aus Relationalen Datenbanken
- Tags and non-visible metadata: Das Datenbanksystem definiert metadaten um dokumente besser gruppieren zu können. Z.B Zeitpunkt der Dokumenterstellung, Datenquelle oder Felder, die in diesem Dokument vorhanden sind. Diese sind ebenfalls wichtig um Abfragen auf der Datenbank zu optimieren, da wir kein zugrunde liegendes Schema haben.
- Directory Hierachies: Baumstruktur der Dokumente, ähnlich zu dateisystemen.
- Können sowohl physische als auch logische Representationen sein

Replikation und Partitionierung

"The NoSQL movement is motivated by horizontal scalability to leverage clusters of commodity hardware and minimization of the impedance mismatch between the data model of the application and the database. Horizontal scalability is addressed through partitioning and replication"

-> Dokumente sind in sich konsistent



Quelle: Felix Gessert, Norbert Ritter: Scalable Data Management: NoSQL Data Stores in Research and Practice

Document Stores - Victor Wolf

19.05.2020

6

Dokumente haben keine Beziehungen die in der Datenbank direkt repräsentiert werden. Es gibt nur indirekte Beziehungen die wir den Dokumenten erst beim Abfragen zuschreiben.

Beispiele für Document Stores



Document Stores - Victor Wolf

19.05.2020

7

CouchDB ist von der apache software foundation.

elasticSearch ist ein spezieller document store sondern eher eine Suchmaschine, die aber auf Dokumenten basiert.

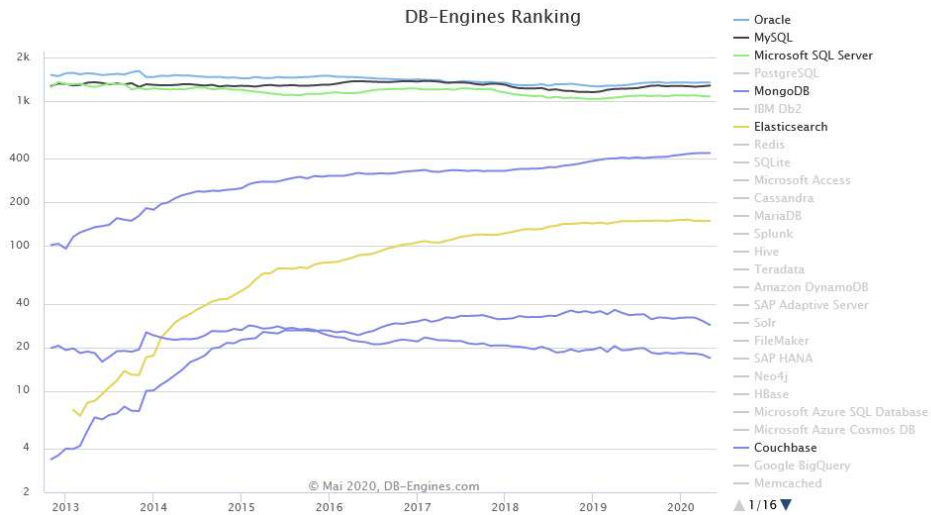
Couchbase von Couchbase Inc hat aber nichts mit CouchDB zu tun

Quiz

Welche von den vier ist der meist benutzte Document Store?

- A: MongoDB
- B: CouchDB
- C: elasticSearch
- D: Couchbase

Quiz



Document Stores - Victor Wolf

19.05.2020

9

MongoDB ist der Spitzenreiter

Gefolgt von Elasticsearch

Gefolgt von Couchbase und CouchDB

Also macht es Sinn sich MongoDB genauer anzuschauen

1. Allgemein

2. MongoDB

3. Live Demo

4. Fragen

5. Key Takeaways



Versprechen

Einfach zu benutzen

Schnell

Skalierbar

Verfügbar

Trotz NoSQL komplexe Datenaggregation

ACID auf Dokumenten Ebene

Document Stores - Victor Wolf

19.05.2020

11

- Das sind die Werbeversprechen auf der MongoDB Seite
- Werden wir jetzt ansehen wie sie das versuchen umzusetzen
- ACID mittlerweile auch auf Operationsebene aber nur wenn man das will.

Documents

```
{  
  name: "sue",  
  age: 26,  
  status: "A",  
  groups: [ "news", "sports" ]  
}
```

← field: value
← field: value
← field: value
← field: value

Dokumente in MongoDB folgen dem BSON Format, welches einfach ein Binarisiertes JSON Format ist.

PLUS das `_id` feld was immer hinzugefügt wird und als Primary Key dient.

Das ist der einzige Primary Key den es geben kann.

API - Abfragesprache

```
mongodb@ubuntu:~/mongodb-linux-x86_64-2.6.0$ bin/mongo
MongoDB shell version: 2.6.0
connecting to: test
> db.adminCommand( { getLog: "global" } )
{
  "totalLinesWritten" : 34,
  "log" : [
    "2014-05-08T01:36:03.034-0400 [initandlisten] MongoDB ... ",
    "2014-05-08T01:36:03.036-0400 [initandlisten] db version v2.6.0",
    "2014-05-08T01:36:03.038-0400 [initandlisten] git version: ... ",
    ...
  ],
  "ok" : 1
}
```

Websocket auf Port 27017 mit binärem Protokoll
MongoDBs integrierte Abfragesprache ist JavaScript
MongoDBs shell ist somit eine interaktive JavaScript Umgebung
Eigentliche Abfragen sind dann JSON

API - Treiber



Document Stores - Victor Wolf

19.05.2020

14

Treiber sind gut um die requests nicht selbst implementieren zu müssen.
Meiste aktuelle Programmiersprachen werden von MongoDB unterstützt.
Selbst Programmiersprachen wie Haskell werden von der Community unterstützt

CRUD Operations - Create

```
db.users.insertOne(  
  {  
    name: "sue",  
    age: 26,  
    status: "pending"  
  }  
)
```

← collection

← field: value
← field: value
← field: value } document

Ein Beispiel für einfügen.

Javascript ist wieder die Sprache.

Eigentliches Objekt was erzeugt werden soll ist in JSON.

Dokument Operationen sind atomic

CRUD Operations - Retrieval

```
db.users.find(  
  { age: { $gt: 18 } },  
  { name: 1, address: 1 }  
).limit(5)
```

← collection
← query criteria
← projection
← cursor modifier

Das Objekt, was wir übergeben ist diesmal ebenfalls JSON auch wenn es aussieht als würden wir dieses Objekt einfügen wollen, übergeben wir einfach eine Operation für die Datenbank in JSON.

Dabei müssen Felder nicht unbedingt Felder eines Dokuments sein sondern können z.B auch Operatoren sein.

Aggregation

Document Stores - Victor Wolf

19.05.2020

17

Jede Stage hat macht einen Schritt zur vollständigen Aggregation hin.
Die Befehle sind dabei JSON
Es gibt auch einen MapReduce Ansatz

Aufgabe

Ich möchte alle Dokumente deren amount Feld größer als 280 ist.

Tipp: Der „größer als“ Operator ist „gt“

```
db.orders.aggregate([
  { $match: { amount: { $gt: 280 } } }
])
```

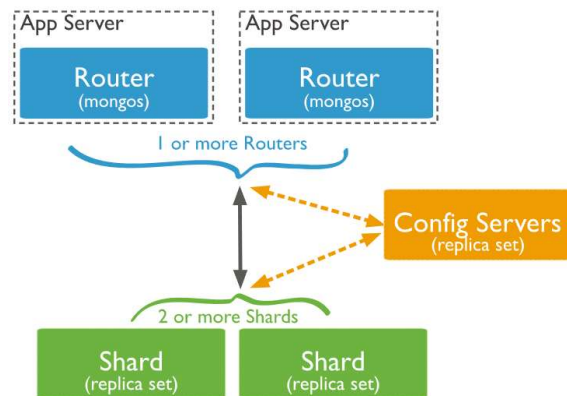
1. \$match 2. amount 3. \$gt 4. 280

Hätte man die Abfrage auch mit einem anderen Keyword einleiten können?

{ cust_id: "A123", amount: 500, status: "A" }
{ cust_id: "A123", amount: 250, status: "A" }
{ cust_id: "B212", amount: 200, status: "A" }
{ cust_id: "A123", amount: 300, status: "D" }

orders

Sharding



Document Stores - Victor Wolf

19.05.2020

19

Shard: Datenteil der aufgeteilten Gesamtdaten. Diese können nochmal repliziert sein. Daten werden pro Collection aufgeteilt. Jeder Shard besitzt einen Shard key in dem steht welche Felder sich in diesem Teil der Daten befinden.

Router: Routet die Anfragen auf die spezifischen Shards. Handelt die Kommunikation mit dem Cluster

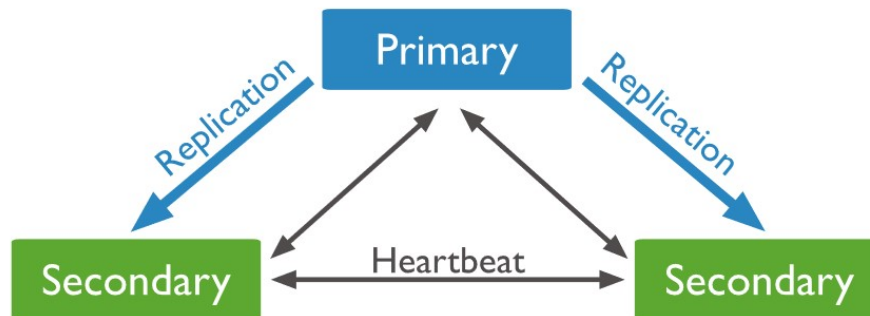
Config Servers: Speichern Metadaten über den Cluster. Welche Collection sind wo?.

Wieviele Shards gibt es?

Wir sehen durch die Documents sind die Daten gut aufteilbar

In der Live Demo alles in einem Computer.

Replikation



Document Stores - Victor Wolf

19.05.2020

20

Gruppe aus mongod instanzen. Mongod ist dabei der Hauptverwaltungsdaemon der MongoDB Datenbank

Primary Node erhält alle schreib Operationen.

Speichert diese in einem Oplog.

Und verteilt dieses oplog weiter zur replikation an die secondaries.

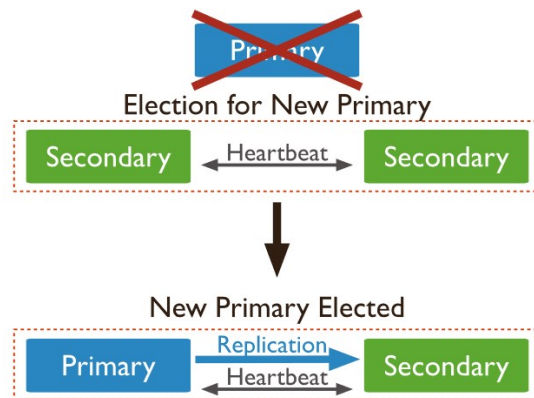
Diese arbeiten das oplog dann asynchron ab.

Wenn ein Replication set erstmals erstellt wird, wird der Datensatz vom Primary geklont.

Die Größe des oplogs ist fest daraus wird abgeleitet ob ein initial sync notwendig ist.

Frage: Was denkt ihr passiert wenn die Primary Node ausfällt?

Replikation



Document Stores - Victor Wolf

19.05.2020

21

Falls die primary node ausfällt haben die Secondaries einen eingebauten timeout, ab wann sie eine neue Wahl zum nächsten primary starten. Dieser Primary übernimmt dann wieder die Write Anfragen

MongoDB vs RDBMS

	MongoDB	RDBMS
Skalierbarkeit	✓	✗
Primär- und Fremdschlüssel	✗	✓
Schnelligkeit	✓	✗
Konsistenz	✗	✓
Atomizität	Auf Dokumentebene	✓
Schema	Kann definiert werden	✓

Gesamte Information für ein Dokument muss in diesem Dokument sein.

1. Allgemein
2. MongoDB

3. Live Demo

4. Fragen
5. Key Takeaways

```
document.getElementById(div).innerHTML = errEmail;
else if (i==2)
{
  div = divs[i];
  var atpos=inputs[i].indexOf("@");
  var dotpos=inputs[i].lastIndexOf(".");
  if (atpos<1 || dotpos<atpos+2 || dotpos>inputs[i].length-1)
  document.getElementById('errEmail').innerHTML = "Invalid email address";
  else
  document.getElementById(div).innerHTML = "Valid email address";
  var atpos=inputs[i].indexOf("@");
  var dotpos=inputs[i].lastIndexOf(".");
  if (atpos<1 || dotpos<atpos+2 || dotpos>inputs[i].length-1)
  document.getElementById('errEmail').innerHTML = "Invalid email address";
  else if (i==5)
  {
    document.getElementById('errEmail').innerHTML = "All emails are valid";
  }
}
```

1. Allgemein
2. MongoDB
3. Live Demo

4. Fragen

5. Key Takeaways

Fragen

Gibt es ein JOIN wie in RDBMS in der MongoDB Abfragesprache?

A: Ja

B: Nein

Nicht ganz:

[https://docs.mongodb.com/master/reference/operator/aggregation/lookup/#pipe._\\$lookup](https://docs.mongodb.com/master/reference/operator/aggregation/lookup/#pipe._$lookup)

Fragen

Was sind die charakterisierenden Eigenschaften eines Dokuments?

- ▶ Eindeutiger Key
- ▶ Strukturierter Text, Objektähnlich
- ▶ Gekapselt

Fragen

Für welche Anforderungen ist MongoDB besonders geeignet?

- ▶ Schnellen Release
- ▶ Unterschiedliche Daten
- ▶ Viele Daten
- ▶ Schnelles Sammeln der Daten wichtiger als schnelle Analyse
- ▶ Webbasiert

- 
1. Allgemein
 2. MongoDB
 3. Live Demo
 4. Fragen

5. Key Takeaways

Key Takeaways Document Stores

Geeignet für viele unterschiedliche Daten

Dokumente als gekapselte objektähnliche Dateneinheit

Schnelligkeit im Sammeln, im Austausch
gegen Schnelligkeit im Analysieren

Quellen

- ▶ <https://db-engines.com/de/ranking>
- ▶ <https://github.com/ramnes/awesome-mongodb>
- ▶ <https://docs.mongodb.com/manual/>
- ▶ <https://blog.panoply.io/couchdb-vs-mongodb>
- ▶ <https://docs.couchbase.com/home/index.html>
- ▶ <https://docs.couchdb.org/en/stable/>
- ▶ <https://www.mongodb.com/nosql-explained/nosql-vs-sql>